(Report about Transaction and Procedure)

1. What is a Transaction?

Transactions are broad actions involving the exchange of value between entities. They are categorized based on payment timing, the parties involved, and the nature of the objective.

Types of transactions :

1. Financial transactions: These are transactions that involve the exchange of money for goods, services, or assets. Examples of financial transactions include buying a product at a store, withdrawing money from a bank account, or investing in a stock.

2. Legal transactions: These are transactions that involve the transfer of legal rights or obligations. Examples of legal transactions include signing a contract, buying or selling property, or transferring ownership of a business.

3. Electronic transactions: These are transactions that are conducted using electronic devices and networks, such as computers, smartphones, and the internet. Examples of electronic transactions include online shopping, online banking, and online investing.

4. Business transactions: These are transactions that are conducted by businesses in the course of their operations. Examples of business transactions include buying raw materials, selling finished products, and providing services to customers.

5. Government transactions: These are transactions that are conducted by government agencies in the course of their operations. Examples of government transactions include collecting taxes, providing services, and making payments to individuals or businesses.

Key Properties of SQL Transactions: ACID

The integrity of SQL transactions is governed by the ACID properties, which guarantee reliable database transactions. These four properties work together to guarantee that the database remains consistent and reliable.

- **Atomicity:** The outcome of a transaction can either be completely successful or completely unsuccessful. The whole transaction must be rolled back if one part of it fails.

- **Consistency:** Transactions maintain integrity restrictions by moving the database from one valid state to another.

- **Isolation:** Concurrent transactions are isolated from one another, assuring the accuracy of the data.

- **Durability:** Once a transaction is committed, its modifications remain in effect even in the event of a system failure.

Example of SQL Transaction with a Bank Transfer Scenario:

```
BEGIN TRANSACTION;

-- Deduct $150 from Account A
UPDATE Accounts
SET Balance = Balance - 150
WHERE AccountID = 'A';

-- Add $150 to Account B
UPDATE Accounts
SET Balance = Balance + 150
WHERE AccountID = 'B';

-- Commit the transaction if both operations succeed
COMMIT;
```

If any error occurs, such as an issue with the UPDATE query, you can use ROLLBACK to undo all changes made during the transaction:

```
ROLLBACK;
```

What is a Stored Procedure?

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.

You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

Stored Procedure Syntax

```
CREATE PROCEDURE procedure_name
AS
sql_statement
GO;
```

Example of SQL(Procedure)

```
-- Create a stored procedure named "GetCustomersByCountry"
CREATE PROCEDURE GetCustomersByCountry
    @Country VARCHAR(50)
AS
BEGIN
    SELECT CustomerName, ContactName
    FROM Customers
    WHERE Country = @Country;
END;

-- Execute the stored procedure with parameter "Sri lanka"
EXEC GetCustomersByCountry @Country = 'Sri lanka';
```

Types of SQL Stored Procedures

1. System Stored Procedures

These are predefined stored procedures provided by the SQL Server for performing administrative tasks such as database management, troubleshooting, or system configuration.

2. User-Defined Stored Procedures (UDPs)

These are custom stored procedures created by the user to perform specific operations. User-defined stored procedures can be tailored to a business's needs, such as calculating totals, processing orders, or generating reports.

3. Extended Stored Procedures

These allow for the execution of external functions, which might be implemented in other languages such as C or C++. Extended procedures provide a bridge between SQL Server and external applications or tools, such as integrating third-party tools into SQL Server.

4. CLR Stored Procedures

These are stored procedures written in .NET languages (like C#) and executed within SQL Server. CLR stored procedures are useful when advanced functionality is needed that isn't easily achievable with T-SQL alone, such as complex string manipulation or working with external APIs.

Advantages of Using SQL Stored Procedures

1. **Improved Performance**: Stored procedures are precompiled, meaning they execute faster than running multiple individual queries.

2. **Enhanced Security**: Users can be granted permission to execute stored procedures without directly accessing the underlying tables.

3. **Code Reusability**: Stored procedures allow for reusability, making it easier to maintain and update code.

4. **Reduced Network Traffic**: By bundling multiple SQL statements into one call, stored procedures reduce network load and improve application performance.

5. **Better Error Handling**: SQL stored procedures provide a structured way to manage errors using `TRY...CATCH` blocks.